Solving stochastic programming models for asset/liability management using iterative disaggregation

Klaassen, P.

1997

Faculteit der Economische Wetenschappen en Econometrie

Serie Research Memoranda

Solving Stochastic Programming Models for Asset/Liability Management using Iterative Disaggregation

Pieter Klaassen

Research Memorandum 1997-10~~680~~

March 1997

vrije Universiteit *amsterdam*

To appear in *World- Wide Asset and Liability Modeling,* W.T. Ziemba and J.M. Mulvey (eds.), Cambridge University Press, 1997.

Solving Stochastic Programming Models for Asset /Liability Management using Iterative Disaggregat ion

Pieter Klaassen

Vrije Universiteit, Financial Sector Management, Department of Economics and Econometrics, De Boelelaan 1105, 1081 HV Amsterdam, and
Rabobank International, Postbus 17100, 3500 HG Utrecht, the Netherlands

1 Introduction

This paper describes a novel approach to solving multiperiod stochastic progamming models for practical portfolio investment problems, called the iterative disaggregation algorithm. A well-known complication of using stochastic programming models in practice is that only a limited amount, of uncertainty can be included, due to the numerical optimization methods which have to be used. An important question is therefore how to choose such a limited description of the uncertainty. On the one hand, the description should be representative of the true uncertainty, while on the other hand one would like to exclude uncertainty which does not affect optimal decisions. Furthermore, one would like to know how sensitive the optimal solution is to the specific choice made. The iterative disaggregation algorithm has been developed with these issues in mind.

The most common method for obtaining an approximate description of the true uncertainty is to randomly sample scenarios from an underlying distribution. Hiller and Eckstein [14], Zenios [26] and Golub *et al.* [10] use sampled interest-rate scenarios in their models for fixed-income portfolio optimization, and exploit the structure of the resulting models by using parallel optimization methods to obtain solutions. Cariño *et al.* [6, 7], Dert [9] and Mulvey and Thorlacius [23] (see also elsewhere in this volume) describe scenario generators for large sets of economic variables, and employ these in multiperiod

models for integrated asset and liability management for pension funds and insurance companies. Cariiio et al. and Dert combine sampling techniques with event trees to model the uncertainty. Many other papers on stochastic programming models for dynamic portfolio investment problems focus on the development of efficient optimization methods, and do not explicitly address the question of how to obtain an approximate description of the true uncertainty. Examples are Bradley and Crane [4], Kusy and Ziemba [22] and Mulvey and Vladimirou [24].

Important properties of any description of the uncertainty in future asset prices and returns are that it is free of arbitrage opportunities and consistent with current market prices. However, these properties often seem to be neglected when stochastic programming models for portfolio investment problems are formulated. As is shown in Klaassen [20], a violation of these properties may lead to optimal portfolios in stochastic programming models which are severely biased towards spurious profit opportunities.

Central to the iterative disaggregation algorithm which we will present are aggregation methods which can be applied to condense a description of the asset-price uncertainty by combining states and time periods in such a manner that the condensed description does not contain arbitrage opportunities or inconsistencies with current market prices if this is true for the original description. Given a detailed description of the uncertainty which is arbitrage-free and consistent with market prices, these aggregation methods can thus be used to arrive at a concise but still arbitrage-free description of the uncertainty on which a stochastic programming model can be based.

The iterative disaggregation algorithm starts with the solution to a small, and therefore relatively easy to solve, stochastic programming model with such an aggregated but arbitrage-free description of the asset-price uncertainty. In each iteration of the algorithm, the description of the uncertainty in the model is refined by reversing one or more of the aggregations that were applied to arrive at the initial model (*disaggregations*), and the model re-optimized. To choose which aggregations to reverse in an iteration, an estimate is made of what additional uncertainty will have the largest impact on the optimal solution, where use is made of optimal solutions found in previous iterations.

In this way, uncertainty is only added to the model in places where it seems to be relevant for the optimal solution. Moreover! the sequence of optimal solutions provides direct insight into the sensitivity of the optimal solution to increases in the level of uncertainty. These are clear advantages over the usual way of solving a stochastic programming model only once, and with a description of the uncertainty that one has to decide on ex-ante.

We will describe the type of multiperiod asset/liability management problems which we will be considering in section 2, and formulate it as a multistage stochastic program. In section 3 we will discuss why it is both reasonable

and important to require that the description of the asset-price uncertainty in this formulation is arbitrage-free, and provide a useful characterization of arbitrage-free asset prices. The aggregation methods are described in section 4, as well as their effect on the stochastic programming formulation. Section 5 discusses the steps in the iterative disaggregation algorithm in more detail, and in section 6 we present the results of the application of the algorithm to a small portfolio insurance problem. Section 7 contains conclusions.

2 Problem Formulation

In this section we describe a multiperiod asset/liability management (ALM) problem, and formulate it as a multistage stochastic programming model.

2.1 Multiperiod Asset/Liability Management

We consider an investor who wants to determine a portfolio investment strategy over time to meet a sequence of liability payments in the future. We assume that the investor can only rebalance his portfolio at a finite number of points in time (trading dates) within a planning horizon of fixed length. Security prices at the initial date are known, but prices and returns at future trading dates are unknown. We approximate this uncertainty by assuming that at each future trading date only one of a finite number of *states of the world* can occur. We can then depict the uncertainty in future security prices and returns in the form of an *event tree.*

As an example, Figure 1 shows a recombining binomial event tree with six periods. The nodes in the tree represent states of the world, and the arcs transitions with positive probability. It is a binomial tree because two states can occur at the end of a period for each given state at the beginning of a period. It is a recombining tree (also called a *lattice)* because states in the inner part of the tree can be visited by multiple paths. This is often assumed to limit the number of different states at each trading date.[1] As the numerical example in section 6 employs a recombining binomial event tree, we will use it for illustrative purposes throughout this chapter. However, our analysis can be extended directly to trees which are not binomial and/or do not recombine.

The investor faces a trade-off between the initial cost of the trading strategy which must enable him to meet his liabilities, and the value of the portfolio which is left at the model horizon (*surplus*). We assume that he can borrow money at intermediate trading dates, but require that the surplus must be

[1] It is not possible to use a recombining event tree if securities with path-dependent payoffs (and thus prices) such as mortgage-backed securities are included in the analysis, as each state at a trading date can in that case only correspond to one path in the tree up to that date.

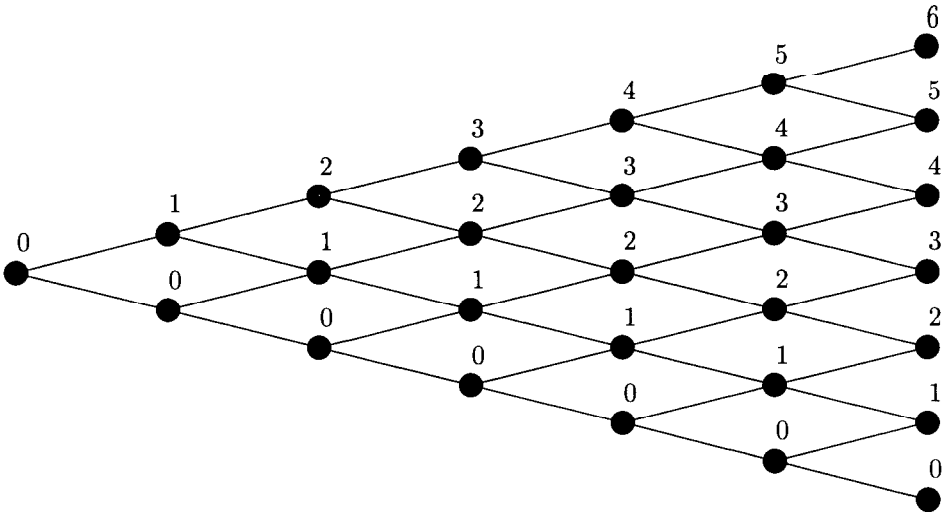


Figure 1: A recombining binomial event tree with six periods.

nonnegative in all cases. In executing a trading strategy, we furthermore assume that the investor has to pay transaction costs, is not allowed to short sell assets, and faces constraints on the amount that he can borrow as well as a spread between the interest rate for borrowing and (risk-free) lending.

The investor's optimal portfolio may be composed differently in different states at a given trading date. Moreover, in the presence of transaction costs, the optimal portfolio composition in a state at a future trading date does in general not only depend on the state itself, but also on the composition of the portfolio that is carried over from the previous period. That is, optimal trading strategies in the event tree are generally *path-dependent*. However, the trading strategies are not allowed to depend on knowledge about the actual course of events in the future. Thus, when two paths in the event tree share the same history up to a certain trading date t, the optimal trading strategy up to time t must be identical on both paths. The trading strategies are then said to be *non-anticipative*.

2.2 Notation

The trading dates are denoted by the index t. The initial date is $t = 0$, the terminal date $t = T$, and intermediate trading dates are $t = 1, \ldots, T - 1$. States of the world are referred to by the index n. Given a state n at trading date $t < T$, each state which can occur with positive probability at time $t + 1$ is called a successor of state n, and will as such be referred to by the index n^+. For any state n at trading date $t > 0$, there is at least one state at time $t - 1$ which has state n as its successor. Such a state is called a *predecessor* of state n, and will be referred to by the index n^-. In the recombining binomial tree of Figure 1, each state at time $t < T$ has two successors, every state in the interior of the tree has two predecessors, while each state on the boundary has

one predecessor. The (subjective) conditional probability of a transition from state n at time t to its successor state n^+ at time $t+1$ is denoted by $\eta_{t/t+1}^{n/n^+}$, and the corresponding probability measure on the event tree by η.

All data in the problem are assumed to be functions of the nodes in the event tree. L_t^n denotes the liability which is due at the end of period t if state n occurs, $D_{i,t}^n$ the dividend payment on security i at the end of period t if state n occurs, and $S_{i,t}^n$ its ex-dividend price in state n at time t. Let I be the total number of securities that is considered by the investor.

The riskless one-period interest rate (continuously compounded and annualized) in state n at time t is r_t^n, and the corresponding discount factor $P_t^n \equiv \exp(-r_t^n \Delta)$, where Δ is the length of a time period in the event tree. P_t^n can be interpreted as the price in state n at time t of a riskless one-period zero-coupon bond that pays one dollar at time $t+1$. The interest-rate spread (continuously compounded and annualized) between the investor's one-period borrowing rate and r_t^n is assumed to be constant through time and denoted by κ. The upper bound on one-period borrowing for the investor in state n at time $t < T$ is written as $\bar{Z}_t^n$.

A scenario s at time t is a path in the event tree between time 0 and time t, and the set of all possible scenarios at time t is denoted by $\mathcal{S}_t$. In the recombining binomial tree of Figure 1, the set $\mathcal{S}_t$ consists of 2^t scenarios. The unconditional probability of scenario s at time t as implied by the probability measure η is denoted by η_t^s.

A scenario $s \in \mathcal{S}_t$ visits one node in the event tree at each trading date between time 0 and time t, and such a node will be referred to by the index $n(s)$. For each scenario s at time t there is exactly one scenario at each time $\tau < t$ which follows the same path in the event tree between time 0 and time τ. This scenario is called the *predecessor* at time τ of scenario s at time t, and will as such be denoted by s^-. Furthermore, each scenario s at time t is the predecessor of one or more scenarios at time $t+1$, which are called *successors* of scenario s. They will be referred to by the index s^+.

Let $\bar{x}_{i,0}$ denote the current holding of security i in the portfolio of the investor (a known number). The variables $xb_{i,t}^s$ and $xs_{i,t}^s$ denote the units of security i that are bought and sold, respectively, in scenario s at time t, and $xh_{i,t}^s$ the holding of security i in scenario s at time t *after* portfolio rebalancing (i.e., the portfolio holdings during period $t+1$). The holding in the riskless one-period security during period $t+1$ if scenario $s \in \mathcal{S}_t$ occurs is denoted by the separate variable y_t^s, while y_T^s denotes the portfolio surplus in scenario $s \in \mathcal{S}_T$. z_t^s represents the amount which has to be repaid by the investor at time $t+1$ due to borrowing in scenario s at time t.

Transaction costs are assumed to be a fraction c of the dollar value traded and apply to both purchases and sales of securities, but not to investment in the risk-free security or borrowing.

2.3 Stochastic Programming Formulation

We capture the investor's trade-off between the initial investment and the value of the portfolio surplus in the objective function: the initial portfolio investment is minimized, but any positive final portfolio value is credited to the objective using a concave utility function $\mathcal{U}(\cdot)$. To be able to make a sensible trade-off, the utility of a surplus must be measured in units of current investment. We assume that the utility function satisfies the expected utility property (see Varian [25, chapter 11]).

The asset/liability management problem can now be formulated as the following multistage stochastic program (each stage corresponds to a time period) :

minimize

$$(1+c)\sum_{i=1}^{I} S_{i,0}xb_{i,0} - (1-c)\sum_{i=1}^{I} S_{i,0}xs_{i,0} + P_0 y_0 - \mathrm{e}^{-\kappa\Delta}P_0 z_0 \quad \sum_{s\in\mathcal{S}_T} \eta_T^s \mathcal{U}(y_T^s)$$

subject to

$$-xs_{i,0} + xb_{i,0} - xh_{i,0} = -\bar{x}_{i,0} \qquad \forall\, i = 1,\ldots,I \quad (2.1)$$

$$xh_{i,t-1}^{s^-} - xs_{i,t}^s + xb_{i,t}^s - xh_{i,t}^s = 0 \quad \forall\, i = 1,..,I, s\in\mathcal{S}_t, t = 1,..,\mathrm{T}\text{-}1 \quad (2.2)$$

$$\sum_{i=1}^{I} D_{i,t}^{n(s)} xh_{i,t-1}^{s^-} + y_{t-1}^{s^-} - z_{t-1}^{s^-} + (1-\mathrm{c})\sum_{i=1}^{I} S_{i,t}^{n(s)} xs_{i,t}^s - (1+\mathrm{c})\sum_{i=1}^{I} S_{i,t}^{n(s)} xb_{i,t}^s$$
$$-P_t^{n(s)} y_t^s + \mathrm{e}^{-\kappa\Delta} P_t^{n(s)} z_t^s = L_t^{n(s)} \quad \forall\, s\in\mathcal{S}_t, t = 1,\ldots,T-1 \quad (2.3)$$

$$\sum_{i=1}^{I}\left(D_{i,T}^{n(s)} + S_{i,T}^{n(s)}\right) xh_{i,T-1}^{s^-} + y_{T-1}^{s^-} - z_{T-1}^{s^-} - y_T^s = L_T^{n(s)} \quad \forall\, s\in\mathcal{S}_T \quad (2.4)$$

$$xs_{i,t}^s, xb_{i,t}^s, xh_{i,t}^s \geq 0 \qquad \forall\, i = 1,\ldots,I,\ s\in\mathcal{S}_t, t = 0,\ldots,T-1 \quad (2.5)$$

$$y_t^s \geq 0 \qquad \forall\, s\in\mathcal{S}_t, t = 0,\ldots,T \quad (2.6)$$

$$0 \leq z_t^s \leq \bar{Z}_t^{n(s)} \qquad \forall\, s\in\mathcal{S}_t,\ t = 0,\ldots,\mathrm{T}\text{-}1 \quad (2.7)$$

The first four terms in the objective function represent the net cost of additional investments at time 0. These additional investments consist of asset purchases (including transaction costs) and investment in the riskless one-period security, while revenues from the sale of assets (net of transaction costs) and borrowing are subtracted. The last term in the objective is the expected utility of a final portfolio surplus.

There are three types of constraints in the model: portfolio-balance constraints, *cash-balance* constraints and *borrowing* constraints. The portfolio-balance constraints link portfolio holdings between successive periods (i.e., before and after rebalancing) for each scenario and each asset. The portfolio-balance constraints are given by (2.1) for all assets at time 0, and by (2.2) for all assets in each scenario after time 0.

The cash-balance constraints make sure that sufficient cash is generated to meet the liability payment in each scenario at each trading date. For each scenario at time $t < T$, this constraint is given by (2.3). At the end of a period, the investor receives dividend payments on his asset holdings and the return on his investment in the one-period riskless security, but has to repay the amount borrowed in the previous period plus interest. This is what the first three terms on the left-hand side of (2.3) represent. The next two terms reflect rebalancing of the portfolio: revenues are generated by selling assets, and money can be invested by buying assets, where both are adjusted for transaction costs. The final two terms on the left-hand side are the investment in the riskless one-period security and the amount borrowed, respectively, during the next period.

The cash-balance constraints (2.4) define the portfolio surplus in each scenario at the terminal date T. The first three terms on the left-hand side determine the final portfolio value before meeting the liability: the portfolio holdings are converted at the appropriate market prices, the return on the investment in the riskless one-period security is added, and the amount due because of borrowing is subtracted. The difference between this portfolio value and the liability payment in a scenario $s \in \mathcal{S}_T$ is the portfolio surplus y_T^s.

The nonnegativity restrictions on $xh_{i,t}^s$ and y_T^s preclude short sales of assets and a negative surplus, respectively, while equation (2.7) limits the amounts which can be borrowed.

3 Arbitrage-free Asset Prices

The formulation of the ALM problem as a stochastic program was based on a description of the uncertainty about future asset prices and returns in the form of an event tree. In this section we discuss why it is both reasonable and important that asset prices in such a description are arbitrage-free, and indicate how it may be constructed using financial asset-pricing models.

3.1 Arbitrage Opportunities and Arbitrage-Free Asset Prices

We speak of an arbitrage *opportunity* if it is possible to construct a self-financing trading strategy (a trading strategy is self-financing if no investments are required after time 0) with payoffs that are nonnegative everywhere and strictly positive in at least one state in the event tree, and for which the initial investment is nonpositive. With such a trading strategy it is thus possible to create something from nothing, and many investors will try to take advantage of that. In fact, there is a large group of investors in today's financial markets, called arbitrageurs, whose main objective is to look for and

exploit arbitrage opportunities. This will influence the prices of the securities involved and lead to the elimination of the arbitrage opportunity. When no arbitrage opportunities exist, asset prices are said to be *arbitrage-free.*[2]

Although arbitrage opportunities may occasionally exist in reality, they are usually short-lived due to the presence of arbitrageurs. Moreover, even if one is able to detect and exploit arbitrage opportunities today, it is preposterous to assume that anyone would be able to foresee their occurrence at future points in time. As the primary objective in asset/liability management is to determine a portfolio investment strategy which forms a robust hedge against the future uncertainty, it furthermore seems imprudent to base such a strategy on the presence of arbitrage opportunities.

In the financial literature, arbitrage opportunities are usually defined for a frictionless world (i.e., a world without transaction costs and taxes, and in which securities are infinitely divisible, interest rates for borrowing and lending are the same, and short sales of assets with full use of proceeds are allowed). In reality, an investor may not be able to exploit such arbitrage opportunities directly because of market imperfections and trading restrictions. Nonetheless, Klaassen [20] illustrates that their presence in a portfolio optimization model with realistic market imperfections and trading restrictions may still significantly bias its optimal solution in unrealistic ways. We therefore require in the sequel that security prices in our ALM model are such that they do not admit arbitrage opportunities in a world without frictions.

Harrison and Kreps [1 1] have derived an important characterization of arbitrage-free security prices in an event tree under the assumption of a frictionless world, which is contained in the following theorem.

Theorem 1 (Harrison and Kreps) Security prices *in an event* **tree are** *arbitrage-free if and only if there exists a positive probability measure on the event tree such that in each given state the expected* one-period *return with respect to this probability measure is identical for all assets.*

The theorem states that there are no arbitrage opportunities in the event tree if and only if there exists a probability measure π such that

$$\frac{\sum_{n+} \pi_{t/t+1}^{n/n+} \left(S_{i,t+1}^{n+} + D_{i,t+1}^{n+} \right)}{S_{i,t}^{n}} = \frac{\sum_{n+} \pi_{t/t+1}^{n/n+} \left(S_{j,t+1}^{n+} + D_{j,t+1}^{n+} \right)}{S_{j,t}^{n}} \qquad (3.8)$$

for all assets i, j and in every state n at each trading date $t = 0,\ldots, T - 1$ in the event tree. The summations in (3.8) are over all successor states $n+$ of state n.

[2] A concept related to arbitrage opportunities is that of "locks" in racetrack betting; see Hausch and Ziemba [12].

Because we have assumed that a riskless one-period security exists in every state in the event tree, this characterization can be restated as

$$S_{i,t}^{n} = P_{t}^{n} \sum_{n^+} \pi_{t/t+1}^{n/n^+} \left(S_{i,t+1}^{n^+} + D_{i,t+1}^{n^+} \right) \tag{3.9}$$

for all assets i in every state n at each trading date $t = 0, \ldots, T - 1$. This equation states that the price of each security i in state n at time t can, under the risk-neutral probability measure π, be written as the expected value of its payoffs at time $t + 1$, discounted by the riskless interest rate.

Because all assets have the same one-period expected return under the probability measure π, this measure is often called *risk* neutral. We emphasize that this risk-neutral probability measure is only a theoretical construct (namely, a necessary and sufficient condition for the absence of arbitrage opportunities), and that it should not be viewed as representing either actual probabilities or subjective probability beliefs of an investor. In other words, whether asset prices in an event tree are arbitrage-free is *independent* of the probabilities assigned to the states in the tree. Thus, an investor may very well believe that the expected return on one asset is higher than that on other assets, which will be reflected by the fact that he assigns probabilities to states (and therefore scenarios) in the tree which differ from risk-neutral probabilities. However, as long as a risk-neutral probability measure can be found he will not able to construct a portfolio which yields a riskless return in excess of the risk-free rate. For more background on the theory of arbitrage-free asset prices, see Huang and Litzenberger [16] and Hull [17].

3.2 Using Financial Asset-Pricing Models

The presence of arbitrage opportunities in portfolio optimization models may stem from several sources. One possible source is that the description of the uncertainty in the model is inconsistent with current market prices, which are usually taken as the prices at which the investor can trade at the initial date. As current market prices of securities are based on expectations of market participants about future prices and cashflows, these expectations should be properly reflected in the event tree which is used as description of the uncertainty in a portfolio optimization model. Another possible source is that prices in the event tree itself are not arbitrage-free. This will be the case, for example, if randomly sampled scenarios from a larger (and possibly arbitrage-free) model are used to construct the event tree.

To construct an event tree in which security prices are arbitrage-free and consistent with market prices, a natural starting point is one of the arbitrage-free asset-pricing models in discrete time which have been proposed in the financial literature. These models are primarily used to calculate prices of derivative securities, and they do so by assuming that arbitrage opportunities

do not exist in financial markets. Examples are the binomial stock-price model of Cox, Ross and Rubinstein [8] and the term-structure models of Black, Derman and Toy [3], Brennan and Schwartz [5], Heath, Jarrow and Morton [13], Ho and Lee [15] and Hull and White [18].

These asset-pricing models, or discretized versions thereof, describe the uncertainty in the underlying variable(s) in the form of a discrete-time, discrete-state event tree. The arbitrage-free value of a derivative security is computed in a recursive manner, starting from the maturity date on which the cash flows (as a function of the value(s) of the underlying variable(s)) are known, and proceeding backwards through time in the tree until the arbitrage-free value at time 0 is found (this recursion employs relation (3.9)). In this way, the security's arbitrage-free value is calculated for every state in the event tree at or before its maturity date.

Although prices of securities in the event tree are therefore arbitrage-free when calculated in this way, there is no guarantee that their arbitrage-free value at time 0 equal current market prices. To accomplish this (at least approximately), one typically selects a set of benchmark securities, and chooses values for the parameters in the model (representing, for example, the volatility of the underlying variable(s)) so that the calculated model prices match the market prices as closely as possible. These parameter values can vary significantly with the number of time steps in the event tree, but they converge to certain limit values when the number of time steps increases.

However, the number of time steps which is necessary to obtain satisfactory convergence is usually much larger than the 10 or at most 20 stages (corresponding to time periods of possibly different lengths) which can be included in a stochastic program[3]. If the number of time steps in the event tree would be restricted to such a small number, the obtained parameter values are typically very sensitive to an increase or decrease in the number of time steps. This implies that the stochastic nature of the underlying variable(s) will not be modelled correctly. Furthermore, the calculated model price of a security which is not included in the benchmark set may vary significantly as a function of the chosen number of steps.

Thus, an event tree that is derived from a financial asset-pricing model in which security prices are arbitrage-free and consistent with current market prices, and which captures the stochastic nature of the underlying uncertainties in a satisfactory way, is typically much too large to include fully in a stochastic program. In the next section we describe aggregation methods which enable a reduction in the size of the event tree without sacrificing its desired properties.

[3] With 10 securities and a binomial event tree with 10 time periods, the ALM model would already have 33,740 variables and 12,266 constraints, and these numbers would approximately double with each additional time period.

4 Aggregation of the Uncertainty

We will combine both states and time periods in a large event tree to reduce its size, and refer to this as *state aggregation* and *time aggregation,* respectively. To illustrate the mechanics of state and time aggregation, we consider the two-period binomial tree of Figure 2a. The vectors at each node in the tree represent the price of a one-period discount bond with a face value of one dollar (P_t^n), the dividend payment on security i $(D_{i,t}^n)$ and the ex-dividend price of security i $(S_{i,t}^n)$. We assume that the prices in the tree are arbitrage-free so that a risk-neutral probability measure π exists on the tree. Let π_t^n denote the conditional risk-neutral probability of an upward transition in the tree from state n at time t. These conditional probabilities are written next to the arcs in Figure 2a. We will reduce this two-period binomial tree through the application of state and time aggregation to a binomial "tree" with only one period. See Klaassen [21] for a complete discussion of the aggregation methods.

4.1 State Aggregation

We will say that *state aggregation* is performed in state n at time t if all successor states of state n in the event tree are combined into one state, the *aggregate* successor. Obviously, the conditional probability of this aggregate successor state is one. Figure 2b shows the resulting event tree when state aggregation is performed in both states at time 1 in the event tree of Figure 2a. The dividends and ex-dividend prices in the aggregate successor states are defined as follows $(n = 0, 1)$:

$$\bar{D}_{i,2}^n \equiv \pi_1^n D_{i,2}^{n+1} + (1 - \pi_1^n) D_{i,2}^n \tag{4.10}$$

$$\bar{S}_{i,2}^n \equiv \pi_1^n S_{i,2}^{n+1} + (1 - \pi_1^n) S_{i,2}^n \tag{4.11}$$

$$\bar{P}_2^n \equiv \pi_1^n P_2^{n+1} + (1 - \pi_1^n) P_2^n. \tag{4.12}$$

Using the no-arbitrage relation (3.9), it follows that

$$\begin{aligned} S_{i,1}^n &= P_1^n \left[\pi_1^n \left(D_{i,2}^{n+1} + S_{i,2}^{n+1} \right) + (1 - \pi_1^n) \left(D_{i,2}^n + S_{i,2}^n \right) \right] \\ &= P_1^n \left(\bar{D}_{i,2}^n + \bar{S}_{i,2}^n \right) \end{aligned} \tag{4.13}$$

which shows that the prices in the aggregated tree of Figure 2b are arbitrage-free.

To obtain Figure 2c from 2b, we perform state aggregation at $t = 0$. Define:

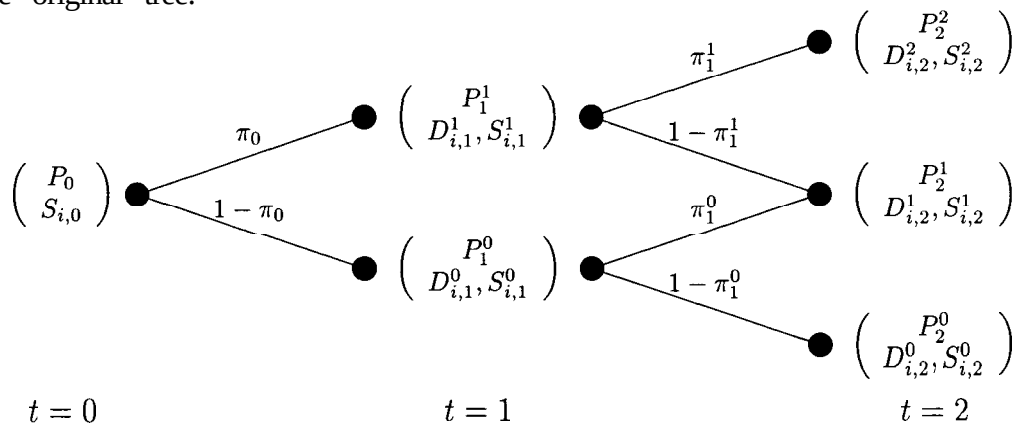
$$\bar{D}_{i,1}^0 \equiv \pi_0 D_{i,1}^1 + (1 - \pi_0) D_{i,1}^0 \tag{4.14}$$

$$\bar{S}_{i,1}^0 \equiv \pi_0 S_{i,1}^1 + (1 - \pi_0) S_{i,1}^0 \tag{4.15}$$

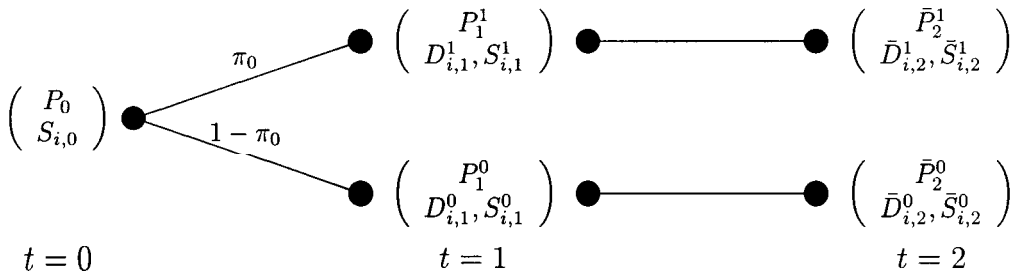
$$\bar{P}_1^0 \equiv \pi_0 P_1^1 + (1 - \pi_0) P_1^0 \tag{4.16}$$

$$\bar{\pi}_1^0 \equiv \left(\pi_0 P_1^1 \right) / \bar{P}_1^0. \tag{4.17}$$

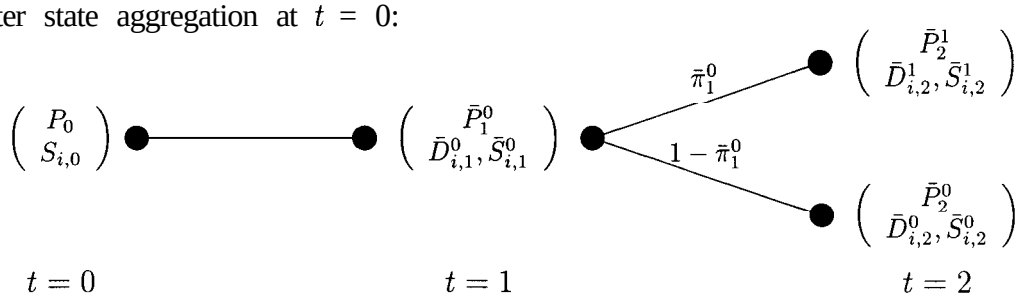
(a) The original tree:



(b) After state aggregation in the nodes at $t = 1$:



(c) After state aggregation at $t = 0$:



(b) After time aggregation at $t = 0$:

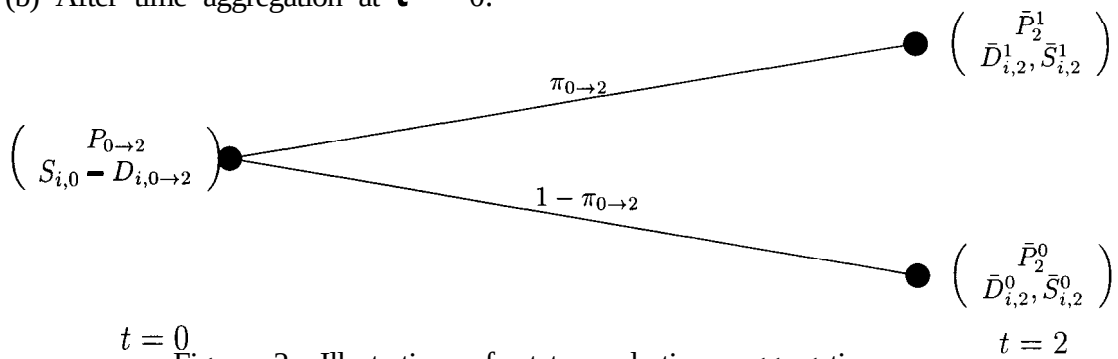


Figure 2: Illustration of state and time aggregation.

Using again the no-arbitrage relation (3.9) yields

$$S_{i,0} = PO\left(\bar{D}^0_{i,1} + \bar{S}^0_{i,1}\right) \tag{4.18}$$

$$\bar{S}^0_{i,1} = \bar{P}^0_1\left[\bar{\pi}^0_1\left(\bar{D}^1_{i,2} + \bar{S}^1_{i,2}\right) + (1 - \bar{\pi}^0_1)\left(\bar{D}^0_{i,2} + \bar{S}^0_{i,2}\right)\right]. \tag{4.19}$$

This shows that prices in the aggregated tree of Figure 2c are arbitrage-free, with $\bar{\pi}^0_1$ acting as risk-neutral probability.

Although state aggregations thus preserve arbitrage-free prices in an event tree, we note that certain relationships between securities with nonlinear payoffs in the original event tree may be violated in an aggregated event tree. For example, suppose the event tree models the evolution of a stock price, and consider a call option on the stock. At its maturity date the option pays the difference between the stock price and its exercise price, if positive, and nothing otherwise. Although this relation will hold exactly in the original event tree, it may be violated in some nodes in an aggregated tree due to the nonlinearity of the option payoff as function of the stock price. Such deviations are unavoidable to maintain arbitrage-free prices for general assets in an aggregated event tree as well as consistency between each asset's payoffs in the tree and its market price.

4.2 Time Aggregation

Time aggregation involves the merging of successive time steps in the event tree. Specifically, time aggregation is performed in state n at time t in an event tree if we replace the transitions from this state to its successors by direct transitions to the successors of its successors. Figure 2d depicts the change in event tree when time aggregation is performed at time 0 in the event tree of Figure 2c.

Time aggregation in Figure 2d eliminates trading date $t = 1$ from the event tree. This affects the definition of a riskless one-period security in the tree as well as the dividend payments. Obviously, dividends which are paid on the trading date that is eliminated cannot be ignored. We will take them into account by assuming that their arbitrage-free value is prepaid at the last trading date in the event tree before the true payment date, i.e., at $t = 0$. The arbitrage-free value (at time 0) of all dividends that are paid on security i between time 0 and time 2, not including the dividends at time 0 and time 2 itself, is denoted by $D_{i,0\to 2}$. Furthermore, the riskless one-period security at time 0 in the event tree is redefined as a security which pays one dollar at time 2, and nothing at other times. Its price is denoted as $P_{0\to 2}$.

Define

$$D_{i,0\to 2} \equiv P_0\bar{D}^0_{i,1} \tag{4.20}$$

$$P_{0\to 2} \equiv P_0\bar{P}^0_1 \tag{4.21}$$

$$\pi_{0\to 2} \equiv \bar{\pi}^0_1 \tag{4.22}$$

and combine this with equations (4.18) and (4.19) to get

$$S_{i,0} = D_{i,0\to 2} + P_{0\to 2}\left[\pi_{0\to 2}\left(\bar{D}^1_{i,2} + \bar{S}^1_{i,2}\right) + (1 - \pi_{0\to 2})\left(\bar{D}^0_{i,2} + \bar{S}^0_{i,2}\right)\right]. \quad (4.23)$$

This relation shows that the price of security i at time 0 can be written as the sum of the present (arbitrage-free) value of its dividend at time 1 and the expected present value of dividend plus ex-dividend price at time 2. As the discounting is performed using the riskless interest rate $(P_{0\to 2})$, the probability that is used to take the expected value $(\pi_{0\to 2})$ is a risk-neutral probability. Because $\pi_{0\to 2}$ is independent of i, the prices in the aggregated event tree of Figure 2d are arbitrage-free.

4.3 The Aggregated ALM model

In the formulation of the ALM model we assumed that in between trading dates no dividends were paid on securities and no liabilities were due. Clearly, we can not assume this any longer if we perform time aggregations in the underlying event tree. We will therefore indicate the changes which have to be made to the formulation in order to take prepayment of the arbitrage-free value of intermediate dividends as well as liabilities into account. This adjusted formulation, which results after performing state and time aggregations in the event tree which underlies the original ALM model, will be called the aggregated ALM model. For its exact mathematical formulation and the precise definition of all aggregated quantities, see Klaassen [21].

Let the trading dates in the aggregated event tree be $t_0, t_1, \ldots, t_T$, with $t_0 = 0$ and $t_T = T$. The period between two successive trading dates in the aggregated tree thus comprises one or more periods in the original tree. In the aggregated ALM model, an extra term is subtracted from the objective function which represents the prepayment of the arbitrage-free value of dividends which are in reality paid between the trading dates t_0 and t_1, not including the dividends on these trading dates itself. These dividends are prepaid over the portfolio holdings of the investor between dates t_0 and t_1, i.e., *after* rebalancing the initial portfolio at time t_0. Furthermore, an extra term is added to the objective function, which represents the arbitrage-free value of the liability payments which are due at dates in between times t_0 and t_1.

The cash-balance constraint for each scenario at times $t_j \in \{t_1, t_2, \ldots, t_{T-1}\}$ is modified as follows. To the left-hand side of the constraint an extra term is added which represents the prepayment of the arbitrage-free value of dividends which are in reality paid between the trading dates t_j and t_{j+1}, conditional on the specific scenario at time t_j. This arbitrage-free value does not include the dividends on the trading dates t_j and t_{j+1}. These intermediate dividends are prepaid over the portfolio holdings of the investor between dates t_j and t_{j+1} in the specific scenario under consideration, i.e., after rebalancing at time t_j. To the liability on the right-hand side of the cash-balance

constraint is added the arbitrage-free value of the liability payments which are due at dates in between times t_j and t_{j+1}, conditional on the scenario at time t_j.

An important property of the aggregated ALM model is that it is neither a restriction nor a relaxation of the original ALM model. This property follows from the fact that the constraint set of **the** ALM model after an aggregation can be obtained from the constraint set of the model before the aggregation by performing both elementary row *and* column operations. We will list the sequence of these operations for both state and time aggregation below. A proof of these results can be found in Klaassen [19].

The fact that the aggregated ALM model is neither a restriction nor a relaxation of the original model indicates that the state and time aggregation methods do not discard part of the uncertainty in the original event tree, but instead condense its description. This contrasts with stochastic programming models in which only a (sampled) subset of scenarios from a large event tree is used as description of the uncertainty.

4.3.1 State Aggregation as Row and Column Operations in the ALM Model

We consider state aggregation in state n at trading date t_j in the event tree. This state aggregation corresponds to the following row and column operations in the ALM model for each scenario s at trading date t_j which visits state n:

- For each successor scenario s^+ at trading date t_{j+1}, multiply its cash-balance constraint, the portfolio-balance constraint of each asset, and the borrowing constraint with the conditional risk-neutral probability $\pi_{t_j/t_{j+1}}^{n(s)/n(s^+)}$. Subsequently, sum the cash-balance constraints of all successor scenarios s^+ at time t_{j+1}, and do the same for the borrowing constraints and each of the portfolio-balance constraints.
- Add the columns which correspond to the portfolio sales variables $xs_{t_{j+1}}^{s^+}$ over all successor scenarios s^+ at trading date t_{j+1} (that is, sum the column coefficients of $xs_{t_{j+1}}^{s^+}$ over all s^+ in each constraint). Repeat this for the variables $xb_{t_{j+1}}^{s^+}$, the variables $xh_{t_{j+1}}^{s^+}$, the variables $y_{t_{j+1}}^{s^+}$, and the variables $z_{t_{j+1}}^{s^+}$.

When $t_{j+1} = t_T$ only those row and column aggregations are performed which are applicable, as there are no portfolio-balance and borrowing constraints in the model for scenarios at time t_T, nor are there variables $xs_{t_T}^s$, $xb_{t_T}^s$, $xh_{t_T}^s$, and $z_{t_T}^s$.

Aggregation of the columns which correspond to variables $y_T^{s^+}$ when $t_{j+1} = t_T$ affects the evaluation of the surplus in the objective function. If the proba-

bilities η_T^s in the objective function are equal to the risk-neutral probabilities, and if the utility function $\mathcal{U}$ has the form

$$\mathcal{U}(y_T^s) = \lambda \left(\prod_{j=0}^{T-1} P_{t_j}^{n(s^-)} \right) y_T^s \tag{4.24}$$

with $\lambda \leq 1$ a scalar, then this aggregation is uniquely defined. In this case the expected utility of a surplus is equal to its expected present value, weighted by the scalar λ, with discounting taking place against the one-period interest rates along the scenario path, and the expectation being calculated under the risk-neutral probability measure. This is also the case that will be considered in the numerical example of section 6, and we will therefore focus our discussion on this case. In other cases, however, one must decide how to define the utility of an aggregated surplus in the objective function (see Klaassen [21]).

If the underlying event tree is recombining, then some scenarios which are distinguishable before the state aggregation may become duplicates of each other after the aggregation. This will be the case for scenarios which visit node n at time t_j, and follow the same path through the tree before the aggregation *except* for the node at trading date t_{j+1}. Obviously, one can combine such duplicate scenarios in the ALM model after the aggregation, and sum their respective probabilities. (This is equivalent to multiplying the duplicated constraints of these scenarios by their relative risk-neutral probabilities, summing the corresponding constraints, and adding the columns of corresponding duplicated variables.)

4.3.2 Time Aggregation as Row and Column Operations in the ALM Model

Consider now time aggregation in state n at trading date t_j. We assume that this state has only one successor state at the next trading date t_{j+1} (compare with Figures 2c and d). Otherwise, we can always first perform a state aggregation in this state to obtain this situation. The following row and column operations are performed in the ALM model for each scenario s in state n at trading date t_j:

- The time aggregation implies that no securities can be bought and sold in the successor scenario s^+ at time t_{j+1}, and thus we essentially add the constraints $xs_{i,t_{j+1}}^{s^+} = 0$, $xb_{i,t_{j+1}}^{s^+} = 0$ and $xh_{i,t_{j+1}}^{s^+} = xh_{i,t_j}^{s}$ for all assets i. Instead of adding them, however, we use these constraints to remove the variables $xs_{i,t_{j+1}}^{s^+}$, $xb_{i,t_{j+1}}^{s^+}$ and $xh_{i,t_{j+1}}^{s^+}$ from the ALM model. This renders the portfolio-balance constraints for the successor scenario s^+ at time t_{j+1} vacuous, and they are therefore removed from the formulation.
- Multiply the cash-balance constraint of the successor scenario s^+ at trading date t_{j+1} by the one-period discount bond price $P_{t_j \to t_{j+1}}^{n}$ in

state n at time t_j. If $t_j = 0$, add this constraint to the objective function. Otherwise, add it to the cash-balance constraint of scenario s at time t_j.

- If the borrowing-lending spread κ is positive, multiply the column that corresponds to the borrowing variable $z^s_{t_j}$ by $\left(e^{-\kappa(t_{j+2}-t_{j+1})}P^{n(s^+)}_{t_{j+1}\to t_{j+2}}\right)$ and add it to the column that corresponds to the borrowing variable $z^{s^+}_{t_{j+1}}$.

5 The Iterative Disaggregation Algorithm

We now describe the iterative disaggregation algorithm in more detail. Assume that at the start of the algorithm one has an event tree as description of the uncertainty in asset prices and returns which is arbitrage-free and consistent with current market prices, for example by using arbitrage-free financial asset-pricing models. Such an event tree will most likely be much too large to include in a stochastic programming model for ALM problems. The aggregation methods from the previous section can then be used to reduce the size of the event tree without introducing arbitrage opportunities or losing consistency with current market prices. In this manner one can obtain a small aggregated ALM model which can be solved relatively easily.

In the extreme, one could continue to perform state and time aggregations in the event tree until nothing more than one "expected-value" scenario remains. Obviously, this is not what one wants. In contrast, it is important to include in the ALM model as much of the relevant uncertainty as possible so that its optimal solution is a good approximation to the optimal solution that would result if one were able to incorporate all uncertainty in the model. By starting with a relatively coarse approximation of the true uncertainty in the ALM model, and iteratively refining this approximation based on the information one gathers from the sequence of solutions, this is exactly what the iterative disaggregation algorithm aims to accomplish.

Each iteration of the iterative disaggregation algorithm consists of the following steps:

1. Disaggregate in the aggregated event tree; i.e., reverse one or more aggregations.

2. Find a feasible solution to the disaggregated ALM model, based on the optimal solution from the previous iteration, and re-optimize the disaggregated model.

We will discuss both steps in the remainder of this section.

5.1 Construction of a Feasible Solution after a Disaggregation

From the analysis in section 4 it should be clear that the ALM model after a disaggregation is performed in the underlying event tree will contain both more variables and more constraints than the ALM model before the disaggregation. Hence, the ALM model after a disaggregation is not just a relaxation of the model before the disaggregation, which implies that the optimal solution to the ALM model before the disaggregation may not define a feasible solution in the ALM model after the disaggregation. Here we will be concerned with the question how to construct a feasible solution for the ALM model after a disaggregation, given an optimal solution to the ALM model before the disaggregation. This analysis will be useful when we consider the question in the next subsection how to decide where to disaggregate in the event tree.

Our starting point for the derivation of a feasible solution is the *fixed-weight solution* that is defined by Zipkin [27, 28] in the context of row and column (dis)aggregations in linear programs, and extended by Birge [1] to stochastic linear programs. This fixed-weight solution is defined for both the primal variables (i.e., corresponding to sales and purchases of assets, portfolio holdings, borrowing and lending) and dual variables (shadow prices on the cash-balance, portfolio-balance and borrowing constraints), and relates directly to the column, respectively row, operations of the corresponding aggregation in the ALM model. We will summarize the results for state and time disaggregation in turn. More details can be found in Klaassen [19].

5.1.1 Fixed-Weight State Disaggregation

We consider the reversal of the state aggregation which was described in section 4.3.1. The fixed-weight solution after the state disaggregation in state n at trading date t_j is defined as follows for each scenario s in this state:

- Primal variables:

  The optimal values for the single successor scenario at trading date t_{j+1} before the disaggregation are taken as initial values for each of the successor scenarios at trading date t_{j+1} after the disaggregation. If the original event tree was not recombining, then no scenarios and thus variables are added at other places in the event tree.

  If the original event tree did recombine, however, single scenarios at trading dates beyond t_{j+1} in the event tree before the disaggregation may have to be splitted into multiple scenarios after the disaggregation (see the discussion in section 4.3.1). Initial values for the variables of these new scenarios are set equal to the optimal values of the variables of the single scenario from which they originate.

- Dual variables:

  Initial values for the shadow prices of the constraints for each of the successor scenarios at trading date t_{j+1} after the disaggregation are set equal to the optimal shadow prices for the single successor before the disaggregation, multiplied by the conditional risk-neutral probability of the corresponding state. When the original event tree was a recombining one, and the state disaggregation leads to the splitting of scenarios at later trading dates in the tree, the shadow prices of the constraints for these new scenarios are initialized at the optimal shadow prices of the constraints for the scenario from which they originate, multiplied by their relative risk-neutral probabilities.

The fixed-weight solution for the primal variables will satisfy all portfolio-balance constraints, but it may violate the cash-balance and borrowing constraints in some scenarios. However, the violated constraints are satisfied on *average.* Moreover, if the upper bound on borrowing is state-independent, the borrowing constraints will be satisfied by the fixed-weight primal solution in all scenarios.

5.1.2 Fixed-Weight Time Disaggregation

We consider the reversal of the time aggregation in state n at trading date t_j which was described in section 4.3.2. For each scenario s in this state, the fixed-weight solution after the time disaggregation is defined as follows:

- Primal variables:

  The variables $xh^{s^+}_{i,t_{j+1}}$, $y^{s^+}_{t_{j+1}}$ and $z^{s^+}_{t_{j+1}}$ for the new successor scenario s^+ at trading date t_{j+1} are initialized at the optimal values of these variables for scenario s at trading date t_j. Furthermore, the variables $xs^{s^+}_{i,t_{j+1}}$ and $xb^{s^+}_{i,t_{j+1}}$ are initialized at zero. The optimal values of the lending and borrowing variables for scenario s at trading date t_j are modified as follows:

$$\begin{aligned} y^s_{t_j} &= P^n_{t_j \to t_{j+1}} \hat{y}^s_{t_j} \\ z^s_{t_j} &= \left(\mathrm{e}^{-\kappa(t_{j+1}-t_j)} P^n_{t_j \to t_{j+1}} \right) \hat{z}^s_{t_j} \end{aligned}$$

  where $\hat{y}^s_{t_j}$ and $\hat{z}^s_t$ are the optimal values of the one-period lending and borrowing variables before the disaggregation.

- Dual variables:

  Initial values for the shadow prices of the cash-balance and portfolio-balance constraints for the new successor scenario at trading date t_{j+1} are set equal to the optimal shadow prices for scenario s at trading date t_j before the disaggregation, multiplied by the discount bond

price $P^n_{t_j \to t_{j+1}}$. For scenario s at trading date t_j itself, the initial shadow prices of these constraints are set equal to the optimal shadow prices before the disaggregation.

For the borrowing constraints of these scenarios, the initial shadow prices ξ are chosen as follows:

$$\xi^s_{t_j} = \begin{cases} \dfrac{\hat{\xi}^s_{t_j}}{e^{-\kappa(t_{j+2}-t_{j+1})} P^{n(s^+)}_{t_{j+1} \to t_{j+2}}} & \text{if } \bar{Z}^n_{t_j \to t_{j+2}} = \dfrac{\bar{Z}^n_{t_j \to t_{j+1}}}{e^{-\kappa(t_{j+2}-t_{j+1})} P^{n(s^+)}_{t_{j+1} \to t_{j+2}}} \\ 0 & \text{otherwise} \end{cases}$$

$$\xi^{s^+}_{t_{j+1}} = \begin{cases} \hat{\xi}^s_{t_j} & \text{if } \bar{Z}^n_{t_j \to t_{j+2}} = \bar{Z}^{n(s^+)}_{t_{j+1} \to t_{j+2}} \\ 0 & \text{otherwise} \end{cases}$$

where $\hat{\xi}^s_{t_j}$ is the optimal shadow price on the borrowing constraint of scenario s at trading date t_j before the disaggregation.

The only constraints which may be violated by the fixed-weight primal solution are the cash-balance constraints of each scenario s in state n at trading date t_j and its successor scenario s^+ at the added trading date t_{j+1}. (The upper bounds on one-period borrowing in an aggregated ALM model are defined in such a way that fixed-weight time disaggregation will never result in a violation of the borrowing constraint.) Furthermore, it is easy to see that these cash-balance constraints can only be violated if prepayment of dividends and/or liabilities occurs in state n at trading date t_j before the time disaggregation.

5.1.3 Modifying Fixed-Weight Solutions to Obtain Feasibility

We have indicated that fixed-weight disaggregation may result in violations of the cash-balance constraints for some scenarios in the ALM model, and also of the borrowing constraints if state disaggregation is performed. If the cash-balance constraint for a scenario s at time t is violated after a fixed-weight disaggregation, an obvious way to make it feasible is by increasing the amount of short-term lending in case of a cash surplus, and the amount of short-term borrowing in case of a cash deficit. In the last case, however, this may lead to a violation of the borrowing constraint in the scenario. Furthermore, additional short-term lending or borrowing in scenario s at time t also increases the amounts of short-term lending, respectively borrowing, in its descendant scenarios if we correct for violations in the cash-balance constraints in these scenarios in the same manner. Thus, even if additional borrowing in scenario s at time t does not violate the borrowing constraint at that time, it may cause a violation of the borrowing constraint in one of its successors.

The upper bounds on short-term borrowing may therefore hinder the construction of a feasible solution from the fixed-weight solution through adjustments of the one-period borrowing and lending amounts. To circumvent

this problem, we can relax the ALM model by including for all scenarios at each trading date an additional variable for short-term borrowing on which no upper bound is imposed. A high borrowing-lending spread will be associated with these additional borrowing variables, however. Furthermore, an extra variable is added for each scenario at the terminal date to take care of a negative final portfolio value. The expected present value of such a final portfolio deficit is added to the objective function, and penalized using a (large) penalty parameter.

As is shown in Klaassen [19, section 4.3], it is possible to derive lower bounds on the borrowing-lending spread and the penalty parameter so that the optimal solution will involve neither borrowing nor final portfolio deficits for any pair of values above these lower bounds. Thus, for any pair of values above these lower bounds it does not matter whether one solves the true ALM model, or its relaxation as just defined. The lower bounds are shown to be positively correlated to the variability of asset returns, and negatively to the value of λ when the utility function is defined according to equation (4.24). For example, if $\lambda = 1$ in equation (4.24), no borrowing will take place in the optimal solution whenever the borrowing-lending spread is positive.

5.2 Choosing Disaggregations

In each iteration of the iterative disaggregation algorithm one has to select one or more states in the event tree in which to perform state and/or time disaggregation. The method we will discuss for doing this is based on the infeasibilities of fixed-weight disaggregation, and estimates the effect of these infeasibilities on the objective function. This method is used in the implementation of the iterative disaggregation algorithm in the next section.

Suppose we perform state disaggregation in state n at trading date t_j in the event tree. For each scenario s in this state the fixed-weight primal solution may violate the cash-balance and the borrowing constraints for its new successor scenarios s^+ at trading date t_{j+1}. Let $U_{t_{j+1}}^{s^+}$ denote the shortfall in the cash-balance constraint for successor scenario s^+ at trading date t_{j+1}, i.e., the difference (if positive) between the liabilities in state $n(s^+)$ and the cash flows from the portfolio as implied by the fixed-weight primal solution. Similarly, let $V_{t_{j+1}}^{s^+}$ denote the amount of one-period borrowing in this scenario which exceeds its upper bound.

We estimate the effect of these infeasibilities on the objective function[4] by the following quantity ε:

$$\varepsilon_{t_j}^n \equiv \sum_{s \in \mathcal{S}_{t_j}^n} \sum_{s^+} \left(\tilde{\varphi}_{t_{j+1}}^{s^+} U_{t_{j+1}}^{s^+} + \tilde{\xi}_{t_{j+1}}^{s^+} V_{t_{j+1}}^{s^+} \right) \qquad (5.25)$$

[4] Strictly speaking, we consider infeasibilities in the formulation of the ALM problem in which the cash-balance constraints are written as greater-than-or-equal-to constraints, as we only take shortfalls and not surplusses into account.

where $\tilde{\varphi}^{s^+}_{t_{j+1}}$ and $\tilde{\xi}^{s^+}_{t_{j+1}}$ denote the shadow prices on the cash-balance and borrowing constraint, respectively, as defined by the fixed-weight dual solution, while $\mathcal{S}^n_{t_j}$ denotes the set of all scenarios which visit state n at trading date t_j.

A similar measure can be defined for time disaggregation in state n at trading date t_j. Fixed-weight time disaggregation may lead to a violation of the cash-balance constraint for a scenario s in this state as well as for its successor scenario s^+ at the newly added trading date (which we denote as t_{j+1} for simplicity). With U^s_t again denoting the shortfall in the cash-balance constraint for scenario s at trading date t, we estimate the effect on the objective function of the infeasibilities due to fixed-weight time disaggregation as:

$$\zeta^n_{t_j} \equiv \sum_{s \in \mathcal{S}^n_{t_j}} \left(\tilde{\varphi}^s_{t_j} U^s_{t_j} + \tilde{\varphi}^{s^+}_{t_{j+1}} U^{s^+}_{t_{j+1}} \right) \tag{5.26}$$

where $\tilde{\varphi}^s_{t_j}$ and $\tilde{\varphi}^{s^+}_{t_{j+1}}$ denote the shadow prices on the cash-balance constraints as defined by the fixed-weight dual solution.

We can calculate the quantities ε and ζ for each state in the event tree in order to decide where to perform a state or time disaggregation. Clearly, the higher a quantity is in a state, the larger the estimated effect of the corresponding disaggregation is on the optimal objective value, and thus the more important this disaggregation is estimated to be.

Another approach to deciding where to perform disaggregations in the event tree is to calculate *bounds* on the possible change in the objective function. Zipkin [28] shows how such bounds can be derived for (dis)aggregations in general linear programs if (generalized) upper bounds on the primal and dual variables are known. For the ALM problem, Klaassen [19] shows how upper bounds on the primal and dual variables can be derived if an upper bound on the investment at time 0 is known.

The difference between the arbitrage-free value of all liabilities and the value of the investor's initial portfolio always forms a lower bound on the optimum objective value. Furthermore, the fixed-weight solution in the relaxation of the ALM model after one or more disaggregations always defines an upper bound. One hopes that the bounds which follow from Zipkin's method will be tighter than these general bounds.

5.3 Termination of the Algorithm

For the decision when to terminate the iterative disaggregation algorithm, we would like to have a measure of how close the current solution is to the true optimal solution, i.e., the solution to the ALM model when all uncertainty is considered. It is not obvious, however, how to construct such a measure. Furthermore, because an aggregated ALM model is neither a restriction nor a relaxation of the unaggregated model, it is impossible to tell precisely how

the optimal solution to an aggregated model relates to the solution in the unaggregated model.

One possibility would be to calculate the bounds of Zipkin on the change in objective value, which were discussed earlier. However, these bounds are most likely too weak to be meaningful if one would translate the optimal solution of an aggregated ALM model to a solution for the unaggregated model by fixed-weight disaggregation, because the unaggregated model is typically very much larger than the aggregated models that are solved in the algorithm. Moreover, the sheer size of the unaggregated ALM model may make it practically impossible to perform this calculation.

A more feasible approach is to base the decision to terminate the algorithm on the results in past iterations. In practical applications, an investor will primarily be interested in the optimal portfolio decisions at time 0. One may therefore decide to terminate the algorithm if these portfolio decisions have remained sufficiently stable in recent iterations.

6 A Numerical Example

We now present the results from the implementation of the iterative disaggregation algorithm for a small asset/liability management problem.

6.1 Problem Statement

Consider an investor with interest-rate exposure in his investment portfolio, who wants to limit the downside risk. More specifically, we assume that the investor owns a zero-coupon treasury bond with a maturity of two years from the current date. Furthermore, he has to make a payment after one year, for which he will have to sell the bond. As he expects that interest rates will fall in the coming year, he wants to hold the two-year bond during the first year, but he also wants to be guaranteed that he can fulfill his obligation after one year. He could realize this guarantee if he could buy a one-year put option on the two-year zero-coupon bond.

Assume that all traded options have a maturity of at most four months. The problem therefore becomes to construct a dynamic trading strategy, involving the traded options, with payoffs that replicate the payoffs of the desired one-year put option. The liabilities in the ALM formulation are thus equal to the difference between the scheduled payment and the value of the discount bond, if positive, and zero otherwise. The investor does not accept shortfalls at the payment date.

The face value of the two-year zero-coupon bond is $1000. Assume that the current zero-coupon yield curve is flat with a (continuously compounded) yield of 8% for all maturities. The current price of the bond is therefore

$852.14, and the forward price of the bond for delivery after one year is $923.12. Assume that the scheduled payment equals $932.35 (101% of the forward price), irrespective of the state of the world. That is, the exercise price of the one-year put option that he wants to replicate is $932.35.

Assume that traded option contracts on the two-year bond are initiated at the beginning of every two-month period, and that the options have an initial maturity of four months. For every option maturity, three call options and three put options are traded, which differ only in their exercise prices (respectively 99.5%, 100% and 100.5% of the forward bond price on the maturity date of the options). Thus, at every point in time the investor can trade in six different put options and six different call options on the two-year bond.

The term-structure model of Ho and Lee [15] is used to model the interest-rate uncertainty in the stochastic programming formulation of this problem[5]. This term-structure model describes the uncertainty in the future term structure of interest rates by means of a binomial lattice (see Figure 1). It is a one-factor model as the evolution of the short-term (one-period) interest rate completely determines the evolution of the whole term structure. As input parameters, the Ho and Lee model requires the number of time steps in the binomial lattice for a horizon of given length (in this case one year), the volatility of the one-period interest rate, and the conditional risk-neutral probability of an upward movement in the lattice. This probability is assumed to be the same in every node in the lattice.

Prices of interest-rate derivative securities (such as bonds and options on bonds) in the event tree can be calculated in the recursive manner which is described in section 3.2. For ordinary bonds, the price at the start of the tree which is derived in this recursive manner is guaranteed to be equal to the current market price in the Ho and Lee model, irrespective of the values of its input parameters. For options and other interest-rate derivative securities, however, this is not the case.

Assume that the market prices of the traded options are consistent with a version of the Ho and Lee model which incorporates 120 time steps, in which the risk-neutral binomial probability is $1/2$, and the volatility of the short-term interest rate 0.7% per year. This number of time steps is large enough so that the calculated option values have converged to at least two-decimal precision with the given parameter values, and the chosen volatility level prevents negative interest rates in the model at any point in time. The theoretical value of the replicated put option at time 0 according to this model is $8.73. The data for the traded options, including their price at time 0 if applicable, are listed in Table 1. The initiation and expiration dates are specified in terms of the time steps in the Ho and Lee model. Because options

[5] Although the simple Ho and Lee model suffices to illustrate the iterative disaggregation algorithm in this example, one may want to use a more sophisticated term-structure model in real-world applications (see the references in section 3.2).

Option number	Initiation date	Expiration date	Strike price	Price at time 0: put	Price at time 0: call
10		20	\$859.26 (99.5%)	\$0.40	\$4.66
11		20	\$863.58 (100.0%)	\$1.76	\$1.76
12		20	\$867.90 (100.5%)	\$4.66	\$0.40
20	0	40	\$870.80 (99.5%)	\$0.76	\$5.03
21	0	40	\$875.17 (100.0%)	\$2.28	\$2.28
22	0	40	\$879.55 (100.5%)	\$5.03	\$0.77
30	20	60	\$882.49 (99.5%)	-	-
31	20	60	\$886.92 (100.0%)	-	-
32	20	60	\$891.36 (100.5%)	-	-
40	40	80	\$894.33 (99.5%)	-	-
41	40	80	\$898.83 (100.0%)	-	-
42	40	80	\$903.32 (100.5%)	-	-
50	60	100	\$906.34 (99.5%)	-	-
51	60	100	\$910.89 (100.0%)	-	-
52	60	100	\$915.44 (100.5%)	-	-
60	80	120	\$918.50 (99.5%)	-	-
61	80	120	\$923.12 (100.0%)	-	-
62	80	120	\$927.73 (100.5%)	-	-

Table 1: Data for traded options on the two-year zero-coupon bond.

10, 11 and 12 were initiated before time 0, no initiation date is specified for them. The exercise price of each option is both given as absolute number and as percentage of the forward bond price (between brackets).

6.2 Disaggregation Strategy

To start the iterative disaggregation algorithm, we have aggregated states and time steps in the Ho and Lee model of the previous section to obtain the aggregated event tree of iteration 0 in Figure 4. This initial event tree has only four different scenarios at $t = 120$, and the corresponding ALM model is thus small and easy to solve. The interest rate ranges between 8.128% per year in the lowest state to 7.872% in the highest state, and the corresponding liability (payoffs on the replicated put option) between \$10.55 and \$8.38. Because trading dates are included in this aggregated event tree for all points in time at which dividends are paid (i.e., options expire) and liabilities are due ($t = 120$), no adjustment for the prepayment of dividends and liabilities is necessary.

To prevent that only state and no time disaggregations are performed in this tree, we have slightly modified the use of the sensitivity measures ε and ζ that were defined in section 5.2. First, we have imposed that a state in an

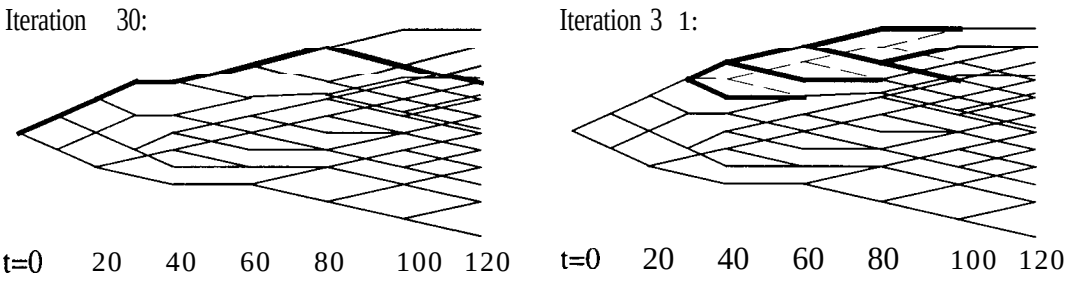


Figure 3: Change in event tree after a state disaggregation on the critical path.

aggregated event tree can never have more than two successor states, and that its successor states occur at the same point in time.

Second, we only calculate the sensitivity measure ε in states that have successor states in which liabilities are specified. In this problem these are the states with successors at $t = 120$. For each of these states we identify a *critical* scenario, which is the scenario with the highest contribution to the sensitivity measure ε in that state. If a state is selected for a state disaggregation based on its value for ε, then a disaggregation is performed along the path in the event tree that corresponds to the critical scenario in that state. If possible, a state disaggregation is performed somewhere along this critical path. If there are multiple possibilities, then the state disaggregation is performed at the earliest point in time at which it is possible. If no state disaggregation is possible, then a time disaggregation is performed in the state at the beginning of the longest period on the critical path (i.e., comprising the largest number of time steps of the Ho and Lee model with 120 time steps). If there is more than one possibility, then the time disaggregation is performed as early as possible in the tree.

We have also used the critical scenario to define the way in which new states after a state disaggregation are connected to the existing event tree. After a state disaggregation is performed in some state along the critical path, then the state disaggregation is basically pushed forward along the path until all new states are connected to the existing tree, or the end of the tree is reached. This is illustrated in Figure 3 for the state disaggregation in iteration 31 of the iterative disaggregation algorithm. The critical path in the event tree of iteration 30 is indicated by the fat line. In iteration 31, a state disaggregation is performed in the state on the path at time 30. The new arcs in the event tree of iteration 31 after the state disaggregation are indicated in bold, and the ones that have disappeared from the event tree are drawn as dashed lines.

A time disaggregation in our implementation consists of both the time and the state disaggregation which were described in sections 5.1.2 and 5.1.1: first a time disaggregation adds a new trading date and a single successor state, and then a state disaggregation in the same state splits the single successor

in two successor states (this corresponds to the change from situation d to situation b in Figure 2). The new trading date is added in the middle between two existing trading dates in the tree.

6.3 Computational Results

We have coded the iterative disaggregation algorithm for this problem on a Sun 10 workstation with 32 MB of internal memory (RAM) in the C programming language. The ALM model has been re-optimized in each iteration as a large linear program, using the CPLEX callable library, where the optimal basis from the previous iteration is used to define a starting basis. The optimal basis columns from one iteration typically do not define a full basis for the model in the next iteration, but CPLEX allows the specification of an incomplete basis, and will complement it with additional columns to construct an initial basis.

The utility function of equation (4.24) with $\lambda = 0.9$ is used to value a portfolio surplus in the objective function, and the scenario probabilities are equal to the risk-neutral probabilities. Assume a transaction cost rate of 1%. The investor can borrow up to \$10 in each state at the riskless one-period interest rate plus one basis point (0.01 percent), and he faces a 1% borrowing spread for any amount in excess of that. Assume that the investor is only allowed to buy options.

Figure 4 depicts the development of the aggregated event tree in the course of the algorithm. Time disaggregations are performed in iterations 17, 26, 30 and 38, and state disaggregations in all other iterations. In the event tree of iteration 40, the interest rate at time 120 decreases from 8.384% at the bottom of the tree to 7.617% at the top. The corresponding payoffs from the replicated put option (the liabilities) range from \$12.82 to \$6.33.

The number of states and scenarios in the event tree increases from 18 and 23, respectively, in iteration 0 to 68 and 342 in iteration 40. The corresponding ALM model has 131 constraints, 485 variables, and 1007 nonzeros in iteration 0 and 1572 constraints, 5120 variables, and 11770 nonzeros in iteration 40. The complete run of 40 iterations only takes a few minutes in real time. The number of simplex pivots required to re-optimize the ALM model in each iteration varies between less than ten and a few hundred. We could not continue the algorithm for many more than 40 iterations before CPLEX required more memory than was available on our computer to perform the re-optimizations.

The optimal value of the objective function in the course of the algorithm is depicted in Figure 5, together with the cost (including transaction costs) of the optimal portfolio at time 0. Both seem to converge in the course of the algorithm. The difference between the two lines represents the value of the portfolio surplus at the terminal date which is credited to the objective. The

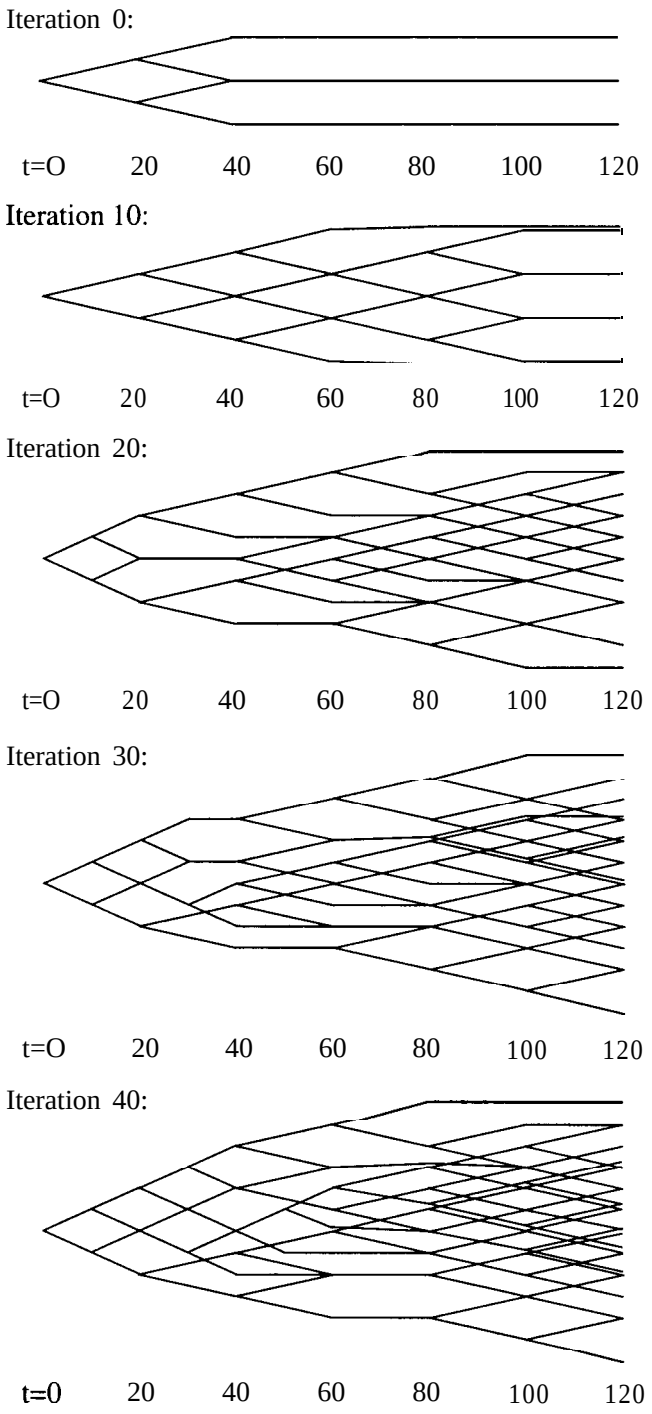


Figure 4: Changes in the event tree during the iterative disaggregation algorithm.

average excess in a scenario at time 120 is \$0.11 with a standard deviation of \$0.28. The standard deviation is relatively high because surplusses almost exclusively occur in the upper part of the event tree (corresponding to low interest rates, and therefore low liabilities).

Figure 6 compares the value of the sensitivity measure with the actual change in the optimum objective value of the ALM model in each iteration. Their correlation is high in early iterations, but decreases later on. A study of the sensitivity measure across different states in each iteration shows that the number of scenarios in a state (and correspondingly, its probability of occurrence) is an important determinant of its value. Our disaggregation strategy thus exhibits a bias towards states in the center of the event tree. As the variability in the number of scenarios per state increases with the growth of the event tree, this bias becomes stronger in the course of the algorithm. This implies that the critical scenario in the state with the largest value of the sensitivity measure plays an increasingly important role for the actual disaggregation that is performed in the event tree.

Typically, interest rates along the critical path decrease initially, and increase after a certain point in time. In the event tree, this corresponds to paths that move upward in the event tree at first, and downward later on. That is, the critical path is initially favorable for the investor, but turns unfavorable after some time. This explains why a majority of the disaggregations are performed in the upper half of the event tree.

The optimal portfolio at time 0 only involves short-term lending and investments in the put options 10 and 20 in every iteration. The selected options are the ones that are most out-of-the-money (i.e., with the lowest strike price). These options provide the investor with the largest relative difference in payoff in different states of the world per option bought. Figure 7 depicts both the optimal amount of short-term lending and the optimal number of options bought in each iteration of the algorithm. Although the option holdings appear somewhat volatile, it should be noted that the dollar amount invested in the options as fraction of the portfolio cost remains fairly constant at a little below 20%. Comparison with Figure 5 shows that the objective value is insensitive to shifts in the option holdings, but that they result in a different trade-off between the initial portfolio cost and the final portfolio value.

6.3.1 Variations in the Transaction Cost Rate

Table 2 shows the optimal portfolio composition at time 0 and the corresponding objective value when the transaction cost rate c varies between 0.1% and 2% (in each case after 40 iterations in the iterative disaggregation algorithm). With higher transaction costs, more of the initial portfolio is invested in short-term lending (which involves **no** transaction costs) and less in the options. A consequence of this change in portfolio composition is that the liabilities are matched less precisely when transaction costs incease.

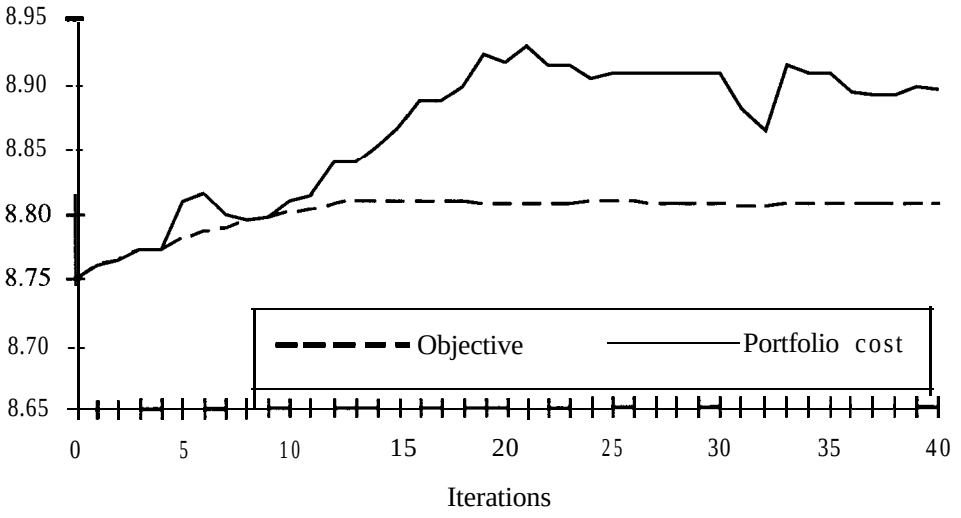


Figure 5: Optimum objective value and initial portfolio cost during the iterative disaggregation algorithm.

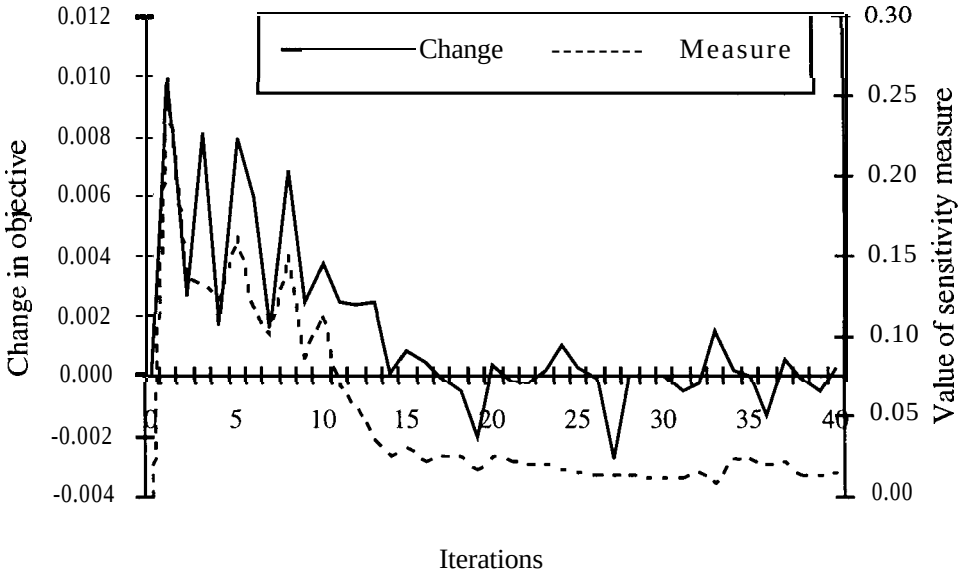


Figure 6: Change in the optimum objective value and the value of the sensitivity measure during the iterative disaggregation algorithm.

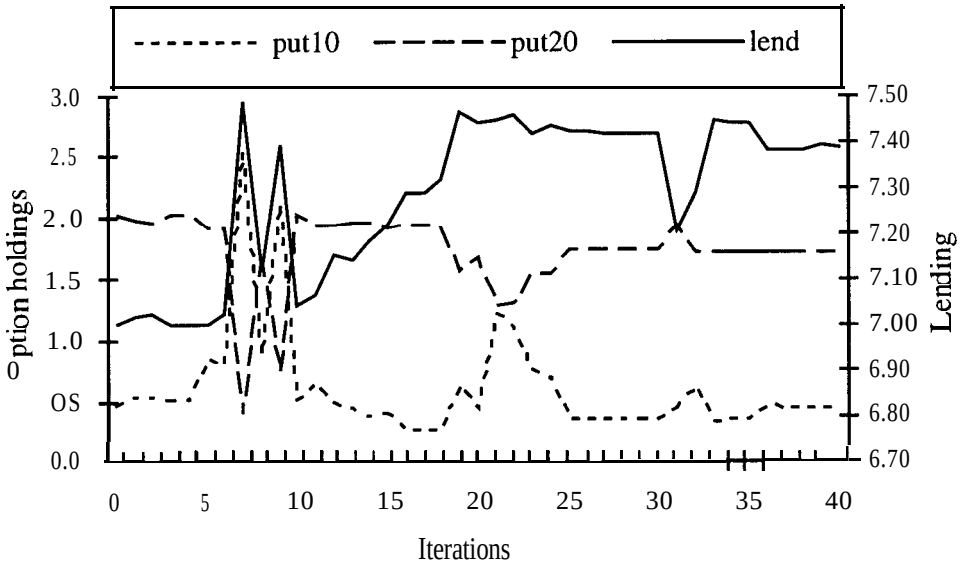


Figure 7: Optimum portfolio composition during the iterative disaggregation algorithm.

c =	Objective	Portfolio cost time 0	Expected final surplus	Portfolio composition		
				put 10	put 20	lending
0.1%	$8.74	$8.74	$0.00	0.872	1.713	$7.08
1.0%	$8.81	$8.89	$0.11	0.450	1.718	$7.39
2.0%	$8.85	$9.18	$0.40	0.118	1.711	$7.80

Table 2: Optimal solution for different transaction cost rates.

The transaction cost rate has a significant impact on portfolio rebalancing after time 0. If the transaction cost rate is 1.0%, additional investments in (out-of-the-money) put options are made after every upward move in the event tree. In contrast, when the transaction cost rate is 2.0%, investments in new options are only made at the expiration date of options in the portfolio, and then only in some of the states of the world.

6.3.2 Variations in the Final-Portfolio Weight

Table 3 displays the optimization results when the final portfolio weight λ varies between 0.8 and 0.98, each time with a transaction cost rate of 1%. When λ increases, it becomes less important to match the liabilities exactly. As a consequence, the transaction cost rate increases in relative importance,

and less portfolio rebalancing takes place. This results in an increase in short-term lending in the initial portfolio, and a significantly higher expected value of the portfolio surplus.

λ =	Objective	Portfolio cost time 0	Expected final surplus	Portfolio composition		
				put 10	put 20	lending
0.80	$8.81	$8.81	$0.00	0.000	2.423	$6.94
0.90	$8.81	$8.89	$0.11	0.450	1.718	$7.39
0.98	$8.77	$10.25	$1.64	0.000	0.084	$10.19

Table 3: Optimal solution for different values of the final-portfolio weight.

7 Conclusions

There are at least two advantages of using the iterative disaggregation algorithm as described in this paper for the solution of stochastic programming models for asset/liability management problems. First, it is based on state and time aggregation methods which can be used to condense a description of the uncertainty in asset prices and returns to any desired level, while being guaranteed that asset prices in this condensed description are arbitrage-free and consistent with current market prices if this was true for the original description. We have discussed why it is both reasonable and important that this description in stochastic programming models satisfies these properties, and indicated how financial asset-pricing models may be employed to construct one.

Second, the algorithm relieves one of the task of having to specify all relevant uncertainty in a stochastic programming model *ex-ante*. Instead, one can start to solve a formulation with a very coarse description of the uncertainty in the extreme, one expected-value scenario and iteratively refine this description based on information that one obtains from solutions in the course of the algorithm. Besides, this sequence of solutions also provides useful insight in the sensitivity of the optimal solution to additional uncertainty in the model. We have illustrated these advantages of the iterative disaggregation algorithm in the numerical example of section 6.

Our description and analysis of the iterative disaggregation algorithm is based on a (arbitrage-free) description of the uncertainty about future asset prices and liabilities in the form of an event tree, but does not depend on assumptions about the nature of the assets or the specific economic factors which influence asset prices and liabilities. In practical applications, however, one usually writes asset returns and liabilities as a function of the development in a set of economic variables. One therefore first needs a description

of the uncertainty in the relevant economic variables before one is able to derive a sensible description of the uncertainty in asset prices and liabilities. This process was illustrated in section 6 with interest rates as the only economic variables and interest-rate derivative securities as the only possible investments. In a real-world setting, more variables will need to be considered (e.g., inflation) so that more asset classes can be included (e.g, inflation-linked bonds and stocks). An important area of research is therefore how a proper and practical description of the joint uncertainty in all relevant economic variables can be obtained, and how a usable, arbitrage-free description of the uncertainty in asset prices and liabilities can be derived from it. Cariño et al. [6, 7], Dert [9] and Mulvey and Thorlacius [23] describe different approaches to this problem.

Several other directions for future research remain. In the implementation of the iterative disaggregation algorithm in section 6, we imposed several restrictions on the disaggregation strategy, concerning for example the number of successor states per state and the trade-off between state and time disaggregations. Furthermore, our disaggregation strategy exhibited a preference for disaggregations in states with a high probability of occurrence, whereas in some situations one may be especially interested in hedging against extreme and unlikely events. Experimentation with different disaggregation strategies, and application to a variety of problems, should provide more insight in the sensitivity of the results to the choice of disaggregation strategy.

In the example of section 6 we re-optimized the ALM model in each iteration as a large linear program, using optimal basis information from the previous iteration. Although this re-optimization was fast, we encountered memory problems due to the size of the stochastic program after a certain number of iterations. This suggests the use of decomposition methods for the re-optimizations. A widely used and generally efficient decomposition method for stochastic programs is Benders' decomposition (see Birge [2]). However, direct application of this decomposition method in the iterative disaggregation algorithm is not very attractive as cuts which are generated in one iteration of the algorithm may not be feasible in the next iteration, due to the fact that the ALM model after a disaggregation is neither a relaxation nor a restriction of the model before the disaggregation (see section 5). Thus, Benders' decomposition is not able to exploit solution information from one iteration to the next. Another decomposition method which does have the ability to exploit such information is the prir -!-dual decomposition method as described in Klaassen [19]. A drawback of this method is, however, that each subproblem must be solved to full optimality, whereas primal-dual methods are known to converge quite slowly to the optimum in general.

The description of the iterative disaggregation algorithm in this paper is based on the formulation of the general ALM problem in section 2, which only includes cash-balance, portfolio-balance and borrowing constraints. In practical applications, one often needs to include additional constraints which

reflect policy or regulatory restrictions. The analysis in sections 4.3 and 5 must then be extended to include these additional constraints. For example, to obtain a feasible solution after a disaggregation when additional constraints are present, one may need to introduce extra slack variables for these constraints and penalize non-zero values of these variables in the objective function, similar to what was done for the borrowing constraints in section 5.1.3. Depending on the specific problem instance, one may also be able to prove that the slack variables are always zero in optimum solutions if the penalties are chosen large enough.

Another area of research is to consider more general utility functions in the formulation of the ALM model to evaluate the portfolio surplus in the objective function. One can preserve linearity of the model by approximating a nonlinear utility function with a piecewise linear function. Alternatively, a generalization of the iterative disaggregation algorithm to convex stochastic programming models must be developed.

The idea of iterative disaggregation of the uncertainty in stochastic programs may be an attractive solution approach for application areas other than asset/liability management as well. In other areas, the restriction of no-arbitrage may not be applicable, which can provide more freedom in how to perform (dis)aggregations. Furthermore, if the stochasticity in the stochastic program is restricted to the right-hand-side vector, Benders' decomposition method can be used efficiently to perform the re-optimizations in the course of the algorithm, as the Benders' cuts from one iteration remain valid for the next iteration.

References

[1] J.R. Birge. Aggregation bounds in stochastic linear programming. Mathematical Programming, 31:25–41, 1985.

[2] J.R. Birge. Decomposition and partitioning methods for multi-stage stochastic linear programs. Operations Research, 33(5):989–1007, Sept.-Oct. 1985.

[3] F. Black, E. Derman, and W. Toy. A one-factor model of interest rates and its application to treasury bond options. *Financial Analysts* Journal, pages 33-39, January-February 1990.

[4] S.P. Bradley and D.B. Crane. A dynamic model for bond portfolio management. *Management Science,* 19(2):139–151, October 1972.

[5] M.J. Brennan and E.S. Schwartz. An equilibrium model of bond pricing and a test of market efficiency. *Journal of Financial and Quantitative Analysis,* 17:301–330, September 1982.

[6] D.R. Cariiio, T. Kent, D.H. Myers, C. Stacy, M. Sylvanus, A.L. Turner, K. Watanabe, and W.T. Ziemba. Russell-Yasuda Kasai model: An asset/liability model for a japanese insurance company using multistage stochastic programming. Interfaces, 24(1):29–49, January-February 1994. Reprinted in this volume, pp. ?? - ??.

[7] D.R. Cariiio, D.H. Myers, and W.T. Ziemba. Concepts, technical issues, and uses of the Russell-Yasuda Kasai financial planning model. Technical report, Frank Russell Company, 1996. Submitted to *Operations Research*.

[8] J.C. Cox, S.A. Ross, and M. Rubinstein. Option pricing: A simplified approach. *Journal of Financial Economics,* 7:229–263, September 1979.

[9] C.L. Dert. *Asset-Liability Management for Pension Funds.* PhD thesis, Erasmus University Rotterdam, September 1995. See also this volume, pp ?? - ??.

[10] B. Golub, M. Holmer, R. McKendall, L. Pohlman, and S.A. Zenios. A stochastic programming model for money management. *European Journal of Operational Research,* 85:282–296, 1995.

[11] M.J. Harrison and D.M. Kreps. Martingales and arbitrage in multiperiod securities markets. *Journal of Economic Theory,* 20:381–408, 1979.

[12] D.B. Hausch and W.T. Ziemba. Locks at the racetrack. *Interfaces,* 20(3):41–48, May-June 1990.

[13] D. Heath, R. Jarrow, and A. Morton. Bond pricing and the term structure of interest rates: A discrete time approximation. *Journal of Financial and Quantitative Analysis,* 25(4):259–304, December 1990.

[14] R.S. Hiller and J. Eckstein. Stochastic dedication: Designing fixed income portfolios using massively parallel benders decomposition. *Management Science,* 39(11):1422–1438, November 1993.

[15] T. Ho and S.-B. Lee. Term structure movements and pricing interest rate contingent claims. *Journal* of Finance, 41(5):1011–1029, December 1986.

[16] C.-f. Huang and R.H. Litzenberger. *Foundations for Financial Economics.* North-Holland, 1988.

[17] J. Hull. *Options, Futures* and *Other Derivative Securities.* Prentice-Hall, second edition, 1993.

[18] J. Hull and A. White. Valuing derivative securities using the explicit finite difference method. *Journal of Financial and Quantitative Analysis,* 25(1):87–99, March 1990.

[19] P. Klaassen. *Stochastic Programming Models for Interest-Rate Risk Management*. PhD thesis, Sloan School of Management, M.I.T., Cambridge, Massachusetts, May 1994. Published as IFSRC Discussion Paper 282-94.

[20] P. Klaassen. Discretized reality and spurious profits in stochastic programming models for asset/liability management. Discussion paper TI 95-73, Tinbergen Institute, Erasmus University Rotterdam, 1995. Accepted for publication in European *Journal of Operations Research.*

[21] P. Klaassen. Financial asset pricing theory and stochastic programming models for asset/liability management: A synthesis. Research memorandum 1996-21, Department of Economics and Econometrics, Vrije Universiteit Amsterdam, 1996. Accepted for publication in *Management Science.*

[22] M.I. Kusy and W.T. Ziemba. A bank asset and liability management model. *Operations Research,* 34(3):356–376, May-June 1986.

[23] J.M. Mulvey and A.E. Thorlacius. The Towers Perrin global capital market scenario generation system. Technical report, Towers Perrin, 1996. See also this volume, pp ?? – ??.

[24] J.M. Mulvey and H. Vladimirou. Stochastic network programming for financial planning problems. *Management Science,* 38(11):1642–1664, November 1992.

[25] H.R. Varian. *Microeconomic Analysis.* Norton, third edition, 1992.

[26] S.A. Zenios. A model for portfolio management with mortgage-backed securities. *Annals of Operations Research,* 43:337–356, *1993.*

[27] P. Zipkin. Bounds on the effect of aggregating variables in linear programs. *Operations Research,* 28(2):403–418, March-April 1980.

[28] P. Zipkin. Bounds for row-aggregation in linear programs. Operations Research, 28(4):903–916, July-August 1980.